BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

57 58

公開情報: (n, E)
検証器を特権化する情報: (ν, δ)
入力: μ
出力: M または invalid

検証器
$C :=$ rand
send (C, n)
receive χ
$\rho := (\chi||\mu)^{\delta} \bmod \nu$
send ρ
receive R
if $R^E \bmod n = C||M$
  output M
else
  output invalid

証明器
receive (C, n)
find (n, t, L)
$f||\nu||c||P_0||P_1||P_2||P_3||P_4||P_5 := L$
if f
  $\chi :=$ rand
  send χ
  receive ρ
  if $\rho^t \bmod \nu = \chi||\mu$
    $M := 0$
    P_0
  else
    $M := 1$
    P_1

    P_2
    if P_3
      P_4
      $R := (C||M)^D \bmod n$
    else
      P_5
      $R :=$ undefind
    send R

【０２１２】［２．１５．第１３の対話プロトコル］（検証器３の構成、図３０）この発明における第１３の検証プロトコルでは、検証器３の特権を与える特徴情報をチケットとして与え、検証器３の特権保証特徴情報による符号化は前述までのチケット証明器２と同様に与える。

【０２１３】図３０はこの検証プロトコルを行なう検証器３の構成図である。

【０２１４】検証器３は特権を保証する秘密情報δによる符号化を行なう場合、チケット保持部１１４から対応するチケットを取りだし、固有情報保持部１１３に保持された固有情報を基に符号化する。

【０２１５】［３．第２の実施例］
［３．１．認証手法］本実施例では、チケットの特徴情報は、以下の形で与えられる。

【０２１６】ｐは素数であり、Ｇは離散対数問題が困難な有限群であり、ｇは有限群Ｇの位数ｐの元であり、ｙ＝ｇ^{x}が満たされるとき、（ｐ、Ｇ、ｇ、ｙ、ｘ）がチケットの特徴情報である。特に、（ｐ、Ｇ、ｇ、ｙ）を公開の特徴情報とし、ｙを秘密の特徴情報とする場合について、実現できるチケットの証明方法について詳述する。

【０２１７】実際には、Ｇを有限体の乗法群として構成したり、有限体上の楕円曲線として構成することができる。

【０２１８】例えば、（ｐ、Ｇ、ｇ）をシステム共通とし、公開情報ｙ、秘密情報ｘとする。チケットはチケット特徴情報ｘと証明器固有情報d_uとチケット付加情報Ｌより計算されるデータであり、証明器２が（Ｃ，ｙ，ｔ，Ｌ）に対して証明器固有情報d_uを用いてτ（Ｃ，ｙ，ｔ，Ｌ）＝C^xが計算できるように構成する。

【０２１９】例えば、d_uをある暗号系の復号鍵とし、E_uをd_uに対応する暗号化とし、Ｈを一方向性関数としたとき、ｔ：＝（E_u（ｘ）、Ｈ（d_u||ｘ||Ｌ））のように定めることができる。

【０２２０】チケットを上述の様に定義すると、チケットを用いた秘密の特徴情報ｘによる符号化C^xは、入力（Ｃ，ｙ，ｔ，Ｌ）に対して証明器２の固有情報d_uによってE_u（Ｄ）を復号して証明器２中でＤを回復し、その値を用いて一方向性関数の値をチケットと比較することによってＬの値の正当性を確認し、回復されたＤの値を用いて証明器２内でC^xを計算するようにすれば良い。

【０２２１】また、d_uを一方向性関数として、ｔ＝ｘ－d_u（Ｌ、ｙ）と定めることもできる。

【０２２２】チケットの上述の定義によれば、入力（Ｃ，ｙ，ｔ，Ｌ）に対してＤによる符号化は$C^{r}C^{d_u(L,r)}$を計算することによって行なうこともできる。

【０２２３】まずは離散対数問題の困難さに安全性の根拠をおく、秘密値共有の原理を利用した対話プロトコルの例を挙げる。

【０２２４】［３．２．第１４の対話プロトコル］（証明器２が時計を持つ定期券型のチケット、図１３、図１９、図２０）第１４の対話プロトコルでは、証明器２の内部状態として時計情報を持っているとする。本プロトコルにおける検証器３の手順を図１３に、証明器２の手順を図１９に従って説明する。

【０２２５】検証器３は、認証情報生成部２８６によって乱数ｒを生成して、Ｃ：＝g^rとしてチャレンジＣを生成して、認証情報保持部２８７に記録する（ステップ１３１）。ステップ1.３２の代わりに、認証情報保持部２８７に記録されたＣと、公開情報ｙをまとめて、送信部２８１より証明器２へ送信する。

【０２２６】ステップ１９１の代わりに、証明器２は、検証器３から送信された（Ｃ，ｙ）を受信部２９２で受信して、情報保持部２９９に記録する。チケット探索部１２２は受信したｙに対応するチケットをチケット保持部１２１から探して、チケットを持っているかどうかを判定する（ステップ１９２）。レスポンスＲにチケットを持たないことを意味する値（この場合は０）を設定して、情報保持部２９９に記録する（ステップ１９７）。ステップ１９２の判定の結果として対応するチケットを持つ場合には、チケットｔ、証明器２の手順$Prover_s$、チケット付加情報Ｌを制御部２９１２内にセットする（ステップ１９３）。

【０２２７】ステップ１９３が実行されたら、ステップ１９４を実行する。この対話の例におけるステップ１９４の詳細な手順を図２０に従って説明する。内部状態保持部１２４の第２の内部状態保持部から読み出したその時の時刻ｔｉｍｅと、Ｌに記述されている有効期限とを比較して、チケットがその時点で有効であるか判定する（ステップ２０１）。チケットを有効であると判定した場合には、第１の演算部２９３においてＲを、Ｒ：＝C^xのように計算して、情報保持部２９９に記録する（ステップ２０２）。特にＲのこの計算を、証明器２の中の第１の演算部２９３ですべて計算してもよいが、チケットｔがｔ＝ｘ－d_u（Ｌ，ｙ）のように構成されている場合には、C^xの計算は$C^{r}C^{d_u(L,r)}$のようにも計算できるので、C^rを証明器２の外で計算して、証明器２の中の第１の演算部２９３で計算した$C^{d_u(L,r)}$とを証明器２の外で掛け合わせても計算できる。証明器２の計算速度が遅い場合には、このような方法も有効になる。ステップ２０１においてチケットを無効であると判定した場合には、必要であるならばチケットをチケット保持部１２１から削除して（ステップ２０３）、Ｒに１を設定して、情報保持部２９９に記録する（ステップ２０４）。ここまでがこの対話の例におけるステップ１９４の詳細な手順である。

【０２２８】ステップ１９４が実行されたら、ステップ１９３でセットされたｔ、$Prover_s$、Ｌを制御部２９１２から解除する（ステップ１９５）。ステップ１９５もしくはステップ１９７が実行されたら、Ｒを送信部２９１より検証器３へ送信する（ステップ１９６）。

【０２２９】検証器３は、受信部２８２において証明器２から送信されたＲを受信して、情報保持部２８９に記録する（ステップ１３３）。y^rを第１の演算部２８３で計算して、その計算結果がＲと一致するかを、正当性検証部２８５で判定する（ステップ１３４）。一致すると判定された場合は証明器２が有効なチケットを持っていることを示すので、「有効」を意味する出力を出力部２８１２より出力する。一致しないと判定された場合は証明器２が有効なチケットを持っていることが示されなかったので、「無効」意味する出力を出力部２８１２より出力する。

【０２３０】以上が第１４の対話プロトコルである。第１４のプロトコルでは、最初の簡単なプロトコルと同様の機能を提供している。

【０２３１】

【表１３】

61　　62

検証器
公開情報: y
出力: '有効' もしくは '無効'
$r := \mathbf{rand}$
$C := g^r$
send (C, y)
receive R
if $y^r = R$

　output '有効'
else
　output '無効'

Prover
if time $< L$
　$R = C^a$
else
　$R = 1$

【０２３２】［３．３．第１５の対話プロトコル］（使い捨て型の回数券型のチケット、図１９、図３２、図３３）第１５の対話プロトコルで利用するチケット補助情報Ｌには、検証器３の公開情報ηと、カウンタi_vの上限値ｉが含まれていることを前提にする。第１５の対話プロトコルにおける検証器３の手順を図３２に、証明器２の手順を図１９にそれぞれ従って説明する。

【０２３３】検証器３は、認証情報生成部２８６によって乱数ｒを生成して、Ｃ：＝g^rとしてチャレンジＣを生成して、認証情報保持部２８７に記録する（ステップ３２１）。認証情報保持部２８７に記録されたＣと、公開情報ｙをまとめて、送信部２８１より証明器２へ送信する（ステップ３２２）。

【０２３４】ステップ１９１の代わりに、証明器２は、検証器３から送信された（Ｃ，ｙ）を受信部２９２で受信して、情報保持部２９９に記録する。チケット探索部１２２は受信したｙに対応するチケットをチケット保持部１２１から探して、チケットを持っているかどうかを判定する（ステップ192）。対応するチケットを持たない場合には、レスポンスＲに０を設定して、情報保持部２９９に記録する（ステップ１９７）。対応するチケットを持つ場合には、チケットｔ、証明器２の手順Ｐｒｏｖｅｒ$_s$、チケット付加情報Ｌをセットする（ステップ１９３）。Ｐｒｏｖｅｒ$_s$のセットとして、Ｌより（η，ｉ）を取り出しておく。

【０２３５】ステップ１９３が実行されたら、ステップ１９４を実行する。本プロトコルにおけるステップ１９４の詳細な手順を図３３に従って説明する。認証情報生成部２９６チャレンジで乱数ｓを生成して、χ：＝g^sとしてチャレンジχを生成して、第２の認証情報保持部２９８に記録する（ステップ３３１）。送信部２９１から検証部２にχを送信する（ステップ３３２）。

【０２３６】検証器３は、χを受信部２８２で受信して、情報保持部２８９に記録する（ステップ３２３）。受信したχが０の場合は、チケットが無くてＲが０として送られたことを示すので、「無効」意味する出力を出力部２８１２より出力して、対話プロトコルを終了する（ステップ３２４）。第２の演算部２８４においてレスポンスρを、

【０２３７】
【数３５】

$$\rho := \chi^r$$

として計算する（ステップ３２５）。計算したρを送信部２８１より証明器２へ送信する（ステップ３２６）。

【０２３８】証明器２は、ρを受信部２９２で受信して、情報保持部２９９に記録する（ステップ３３３）。第２の演算部２９４でη^sを計算して、受信したρと一致するか判定する（ステップ３３４）。判定の結果として一致する場合には、内部状態保持部１２４の第２の内部状態保持部に確保されているカウンタi_vの値をインクリメントする（ステップ３３５）。ステップ３３５が実行されたら、i_vの値をＬより得られた上限値ｉと比較する（ステップ３３６）。ステップ３３７の比較の結果として、i_vの値がｉ以下の場合は、第１の演算部２９３でＲ：＝C^xを計算して、情報保持部２９９に記録する（ステップ３３７）。ステップ３３７の比較の結果として、i_vの値がｉより大きい場合は、内部状態保持部１２４の第２の内部状態保持部からi_vを解放して、チケット保持部１２１からチケットを削除して（ステップ３３８）。ステップ３３４で一致しなかった場合と合

63

わせでＲに１を設定して、情報保持部２９９に記録する（ステップ３３９）。ここまでが第１５の対話プロトコルにおけるステップ１９４の詳細な手順である。

【０２３９】ステップ１９４が実行されたら、ステップ１９３でセットされたｔ、Ｐｒｏｖｅｒｓ、Ｌを解除する（ステップ１９５）。ステップ１９５もしくはステップ１９７が実行されたら、送信部２９１から検証器３へＲを送信する（ステップ１９６）。

【０２４０】検証器３は、受信部２８２において証明器２から送信されたＲを受信して、情報保持部２８９に記録する（ステップ３２７）。y^rを第１の演算部２８３で計算して、その計算結果がＲと一致するかを、正当性＊

64

＊検証部２８５で判定する（ステップ３２８）。一致すると判定された場合は証明器２が有効なチケットを持っていることを示すので、「有効」を意味する出力を出力部２８１２より出力する。一致しないと判定された場合は証明器２が有効なチケットを持っていることが示されなかったので、「無効」意味する出力を出力部２８１２より出力する。

【０２４１】以上が第１５の対話プロトコルである。本プロトコルは、第１の実施例の第３の対話プロトコルと同様の機能を提供する。

【０２４２】

【表１４】

検証器

$r := \mathbf{rand}$

$C := g^r$

send (C, y)

receive χ

$\rho := \chi^{\epsilon}$

receive R

if $y^r = R$

　output '有効'

else

　output '無効'

Prover

$s = \mathbf{rand}$

$\chi := g^s$

send χ

receive ρ

if $\eta^s = \rho$

　$i := i + 1$

　$R = C^s$

else

　$R = 1$

【０２４３】［４．第３の実施例］

（チケットの初期化、図５～図１２）この第３の実施例において、チケット作成・発行装置１とチケット証明器２は、チケットの多重初期化を回避するために、チケットカウント手段i_vを設ける。

【０２４４】チケット作成・発行装置１においては、チケットカウント手段は証明器固有情報保持手段（データベース）７に含ませる構成にしても構わない。また、チケット証明器２においては、チケットカウント手段は内容の書換えが行なえないような防御機構を設けてあればより適切である。

【０２４５】なお、チケット作成・発行装置１によるチケット作成・発行の全体の動作については第４の実施例を参照して後に詳述する。

【０２４６】チケット発行の手順を図５を参照して説明する。図５において、チケット発行時、ステップ５５において、そのチケットが証明器２内部の状態を初期化すると判定されれば、チケット作成・発行装置１のチケットカウント手段からカウンタ値を取り出してチケットと共に送る。初期化が不要な場合はｉ＝０を送る。チケットカウンタ値は最初に１に設定されている。ステップ５６においては次回のチケット発行要求に備えるためにカウンタ値を１増加する。チケットは公開情報だが完全性を保つ必要はある。したがって、センターが署名手段を持ち、証明器識別子とカウンタ値とチケット識別子と初期化手順と証明手順とチケット付加情報とに署名を施すのは好適である。

【０２４７】図６は、発行されたチケットの証明器２へ

の登録を行なう手順を表している。図６において、ステップ６１でチケット保持部に余裕があることを確認し、ステップ６２でチケットへの署名の検証を行なう。ステップ６３で証明器識別子が正しいことを確認し、ステップ６４でｉの値により初期化が必要かどうかを判定する。初期化が必要と判定されるとステップ６５でｉの値が初期化の必要なチケットとしてはじめて現れたかどうかを判定する。

【０２４８】この判定の手順は図７で説明される。はじめてのチケット登録と判定されればステップ６６で初期化が実行される。初期化の実行例は図８および図９で例を挙げて説明する。初期化が不要な場合はチケット保持手段に入力されたチケットを登録する。

【０２４９】図７は、内部状態多重初期化回避の判定を行なう手順を表している。ここでは、証明器２の内部状態保持手段には通し番号のリストを保持する手段を設ける構成として説明する。

【０２５０】チケットカウント手段に保持されたｉ$_{u}$は、今までに初期化が行なわれたチケットの通し番号ｉのうち最大のものであり、内部状態保持手段に保持されたリストはｉ$_{u}$より小さい通し番号のうち未だ初期化されていないものからなる。

【０２５１】入力された通し番号ｉに対して、まずステップ７１でチケットカウント手段に保持されたｉ$_{u}$との比較を行なう。ｉ＞ｉ$_{u}$であれば通し番号ｉに対応する初期化はまだ行なわれていないので、ステップ７２へ進み、そうでないならばステップ７４へ進む。ステップ７２では、ｉ$_{u}$より大きくｉより小さい初期化が未だ行なわれていない通し番号をリストに加える。もしも、リストに加えることができなければリストのオーバーフローなので例外処理を行なう。ついでステップ７３では、ｉ$_{u}$の値をｉで置き換えて、ＯＫを出力する。

【０２５２】ステップ７４では、ｉが内部状態保持手段に保持されたリストに存在するかどうかを判定する。リストにあれば、ステップ７５でｉをリストから削除し、ＯＫを出力する。リストになければ、ＮＧを出力する。

【０２５３】図８と図９はチケットに対する内部状態の初期化の手順を表している。図８および図９において、まず、ステップ８１、９１で初期化の領域確保が可能かどうか判断し、可能であれば、ステップ８２または９２において内部状態ｉ$_{n}$をゼロまたはＬに初期化する。

【０２５４】また、証明器２からのチケットの削除は図１０に示すようにパスワードによる認証で行うことができる。また内部状態の開放も図１１に示すように同様にパスワードによる認証で行うことができる。

【０２５５】［第４の実施例］

（チケット作成・発行装置１のチケット作成・発行、図２、図３、図４、図５）第４の実施例としてチケット作成・発行装置１を説明する。図４はチケット作成の手順を示している。

【０２５６】チケット作成手段４は、チケットの仕様を指示したチケット作成依頼を受け取る。ここでチケットの仕様とは、チケットの検証手順と証明手順と初期化手順とチケット付加情報のデータの型とチケット発行依頼者の資格情報とからなる。

【０２５７】手順の指定には手順に対する識別子を用意して、その識別子で指示しても良いし、手順自体を与るようにしても良い。以下では、仕様の識別子によって指定する方法を述べる。また、チケットが内部状態を持ち、特権を持つ検証者がその内部状態を変更できる場合は、検証手順と証明手順を決定するためには、検証者の特権に対応する証明証も与える必要がある。

【０２５８】図４において、チケット作成手段は、チケット作成依頼を受けるとステップ４１において仕様を検索する。ついで、ステップ４２においてチケットの特徴情報を生成する。ここでは、ＲＳＡ公開鍵暗号系の公開鍵ペアを特徴情報とするものとする。チケットの特徴情報はチケットの作成依頼に応じて作成しても良いし、予め作成して用意しておいても良い。ステップ４３において特徴情報と仕様の識別子とチケット付加情報のデータの型とチケット発行依頼者の資格情報（と必要ならば特権を持つ検証者の証明証）の組をチケット原型データベース６に記憶する。ステップ４４においてチケット識別子ｎとチケット公開情報Ｅとチケット仕様Ｓに対応するチケット検証手順Ｖｅｒｉｆｉｅｒ$_{s}$に対して署名を行ない、それをチケット作成依頼者に与える。

【０２５９】図５はチケット発行の手順を表している。図５において、チケット発行依頼者は、チケット識別子と証明器識別子とチケット仕様識別子とチケット付加情報により、チケットを指定して発行の依頼を行なう。この際、チケット発行手段５はチケット発行依頼者の資格をチケット原型データベースに記憶された資格情報にてらして確かめる。チケット発行手段５は、まず、ステップ５１において証明器識別子に対応する固有情報を検索する。ステップ５２においてチケット識別子に対応するチケットの原型を検索する。ステップ５３において仕様識別子に対応する仕様を検索する。ステップ５４において与えられたチケット付加情報の型が正しいかどうかを判定する。ステップ５５においてチケットが初期化を伴うかどうかを判定する。チケットが初期化を伴うならば、ステップ５６において、証明器固有情報データベース７に記憶されたｉ$_{u}$をｉにセットし、ステップ５７においてｉ$_{u}$をインクリメントする。ステップ５５においてチケットが初期化を伴わないと判定されれば、ステップ５８においてｉを０にセットする。ステップ５９において、今までの実施例で述べたようにＤ，ｄ$_{u}$，Ｌよりチケットｔを生成する。ステップ５１０において証明器識別子Ｕと初期化を伴うチケットの通し番号ｉとチケットの初期化手順Ｉｎｉｔ$_{s}$とチケットの証明手順Ｐｒｏｖｅｒ$_{s}$とチケット付加情報Ｌとチケットｔとの組にチ

ケット発行手段の署名をつけて発行を行なう。

【０２６０】

【発明の効果】これまで述べたように、電子チケットを実現するためには、以下の機能を満たす必要がある。第１点は、正当な権利を持たないものが、不当にチケットを利用することを防止する機能である。第２点は、利用者が自分の保持するチケットの正当性を確認できる機能である。第３点は、当事者間で争いが生じたときに備えて、チケットに与えられた権利内容を第三者に証明できる機能である。さらに状況に応じては、第４点として利用者の匿名性が保証される必要がある。

【０２６１】本発明によれば、複製が非常に困難な証明器とそれに対応するチケットがない限り不正利用が不可能であり、第１の機能を満たす。また、誰が知っていても問題のない公開された情報だけで、利用者の持つチケットの内容を証明できるので、第２および第３の機能を満たす。さらに、チケットの検証時に利用者に依存する情報がまったく通信されないので、利用者の匿名性が維持されて、第４の機能を満たす。このように、本発明によれば、以上の４点すべての機能を満たしたチケットの作成・発行・利用システムが実現される。加えて、チケット発行時および検証時の情報の通信をすべて開示できるので、利用者がその内容を確認することで、利用者の権利を侵害する通信が行われていないことを証明する効果も併せ持つ。

【図面の簡単な説明】

【図１】　この発明の原理的な構成例を示すブロック図である。

【図２】　チケット作成・発行装置の構成を示すブロック図である。

【図３】　チケット作成・発行装置の他の構成を示すブロック図である。

【図４】　チケット作成・発行装置の動作を説明するフローチャートである。

【図５】　チケット作成・発行装置の動作を説明するフローチャートである。

【図６】　証明器へのチケット登録の動作を説明するフローチャートである。

【図７】　多重初期化の判定ルーチンを説明するフローチャートである。

【図８】　証明器の内部状態初期化の一例を説明するフローチャートである。

【図９】　証明器の内部状態初期化の一例を説明するフローチャートである。

【図１０】　証明器からチケットを削除する手順を説明するフローチャートである。

【図１１】　証明器から内部状態を開放する手順を説明するフローチャートである。

【図１２】　チケット検証器およびチケット証明器の構成を示すブロック図である。

【図１３】　チケット検証の手順を説明するフローチャートである。

【図１４】　チケット検証の手順を説明するフローチャートである。

【図１５】　チケット検証の手順を説明するフローチャートである。

【図１６】　チケット検証の手順を説明するフローチャートである。

【図１７】　チケット検証の手順を説明するフローチャートである。

【図１８】　チケット検証の手順を説明するフローチャートである。

【図１９】　チケット証明の手順を説明するフローチャートである。

【図２０】　チケット証明手順の一部を説明するフローチャートである。

【図２１】　チケット証明手順の一部を説明するフローチャートである。

【図２２】　チケット証明手順の一部を説明するフローチャートである。

【図２３】　チケット証明手順の一部を説明するフローチャートである。

【図２４】　チケット証明手順の一部を説明するフローチャートである。

【図２５】　チケット証明手順の一部を説明するフローチャートである。

【図２６】　チケット証明手順の一部を説明するフローチャートである。

【図２７】　チケット証明手順の一部を説明するフローチャートである。

【図２８】　検証手順実行手段の構成を示すブロック図である。

【図２９】　証明手順実行手段の構成を示すブロック図である。

【図３０】　チケット検証器の構成を示すブロック図である。

【図３１】　チケット証明手順を説明するフローチャートである。

【図３２】　チケット検証の手順を説明するフローチャートである。

【図３３】　チケット証明手順を説明するフローチャートである。

【符号の説明】

１　チケット作成・発行装置
２　チケット証明装置
３　チケット検証装置
４　チケット作成手段
５　チケット発行手段
６　チケット原型データベース
７　証明器固有情報データベース

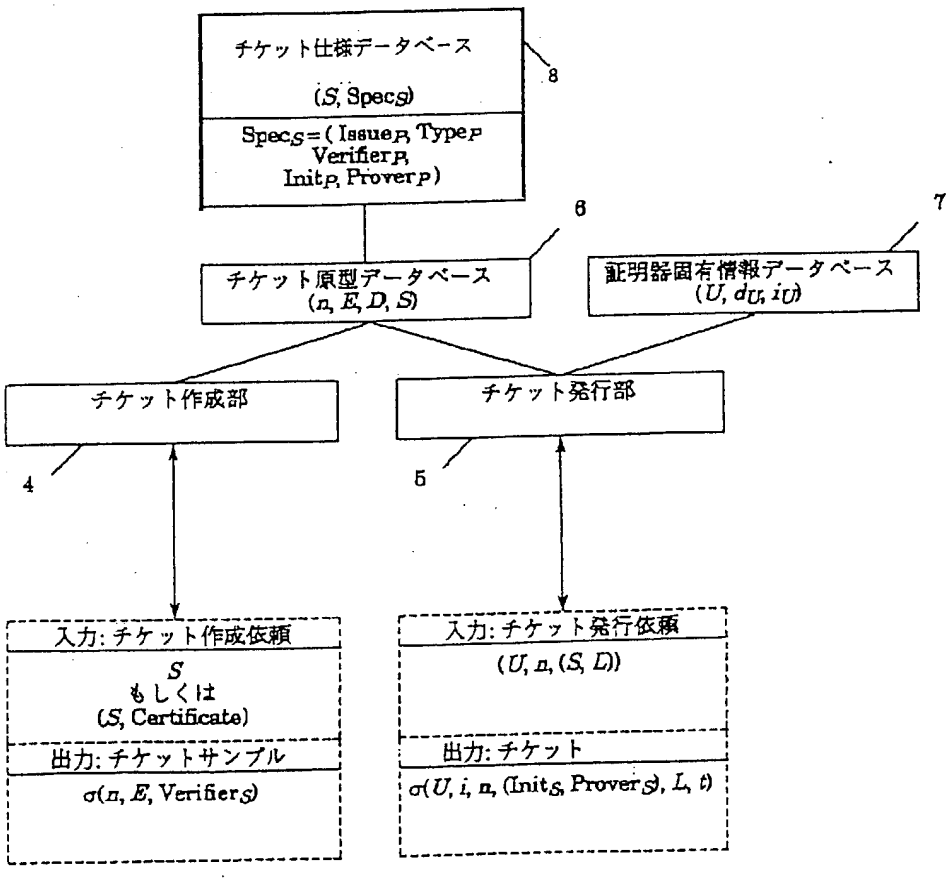
69

１１１　検証手順実行部
１１２　通信部
１２１　チケット保持部
１２２　チケット検索部
１２３　固有情報保持部
１２４　内部状態保持部
１２５　証明手順実行部
１２６　通信部
２８１　検証手順実行部１１１の送信部
２８２　検証手順実行部１１１の受信部
２８３　検証手順実行部１１１の第１の演算部
２８４　検証手順実行部１１１の第２の演算部
２８５　検証手順実行部１１１の正当性検証部
２８６　検証手順実行部１１１の認証情報生成部
２８７　検証手順実行部１１１の認証情報保持部
２８８　検証手順実行部１１１のチケット識別子保持部
２８９　検証手順実行部１１１の情報保持部
２８１０　検証手順実行部１１１の出力情報保持部

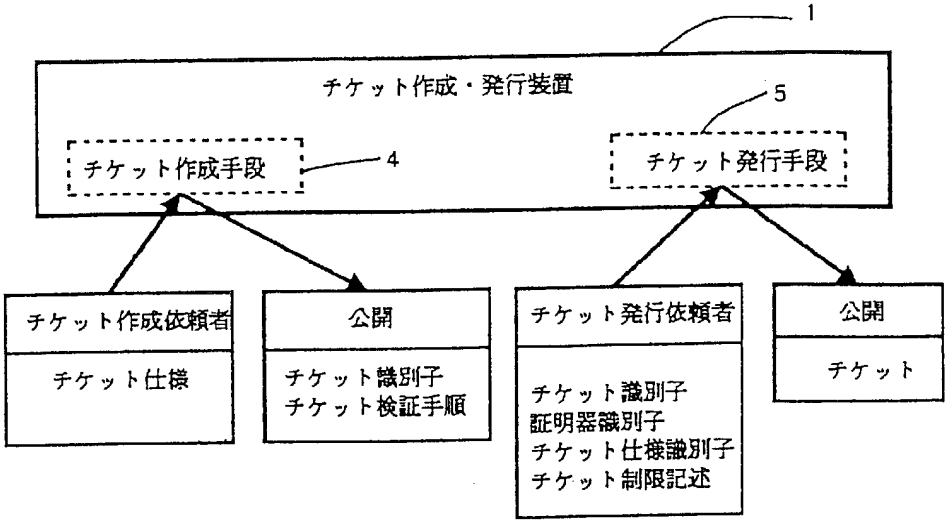
70

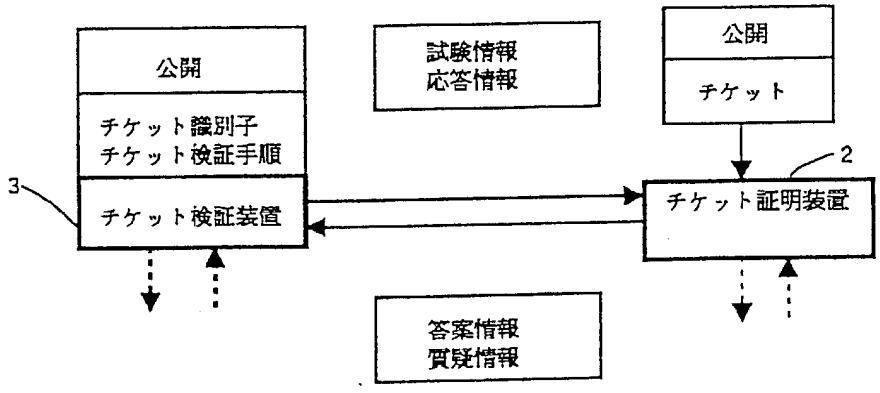
２８１１　検証手順実行部１１１の入力情報保持部
２８１２　検証手順実行部１１１の出力部
２８１３　検証手順実行部１１１の入力部
２８１４　検証手順実行部１１１の利用証拠情報保持部
２８１４
２９１　証明手順実行部１２５の送信部
２９２　証明手順実行部１２５の受信部
２９３　証明手順実行部１２５の第１の演算部
２９４　証明手順実行部１２５の第２の演算部
２９５　証明手順実行部１２５の正当性検証部
２９６　証明手順実行部１２５の認証情報生成部
２９７　証明手順実行部１２５の認証情報保持部
２９８　証明手順実行部１２５の第２の認証情報保持部
２９９　証明手順実行部１２５の情報保持部
２９１０　証明手順実行部１２５の出力情報保持部
２９１１　証明手順実行部１２５の入力情報保持部
２９１２　証明手順実行部１２５の制御部

【図３】

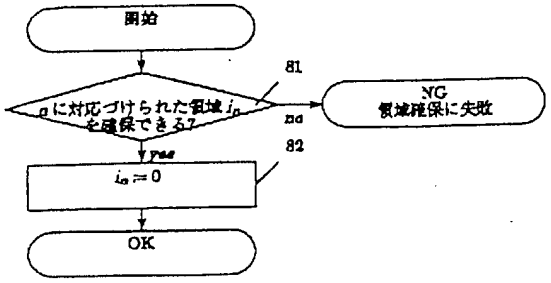


【図１】

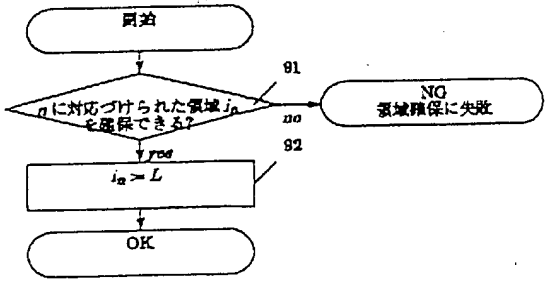

1
チケット作成・発行装置
5
チケット作成手段
4
チケット発行手段
チケット作成依頼者
チケット仕様
公開
チケット識別子
チケット検証手順
チケット発行依頼者
チケット識別子
証明器識別子
チケット仕様識別子
チケット制限記述
公開
チケット


公開
チケット識別子
チケット検証手順
3
チケット検証装置
試験情報
応答情報
公開
チケット
2
チケット証明装置
答案情報
質疑情報

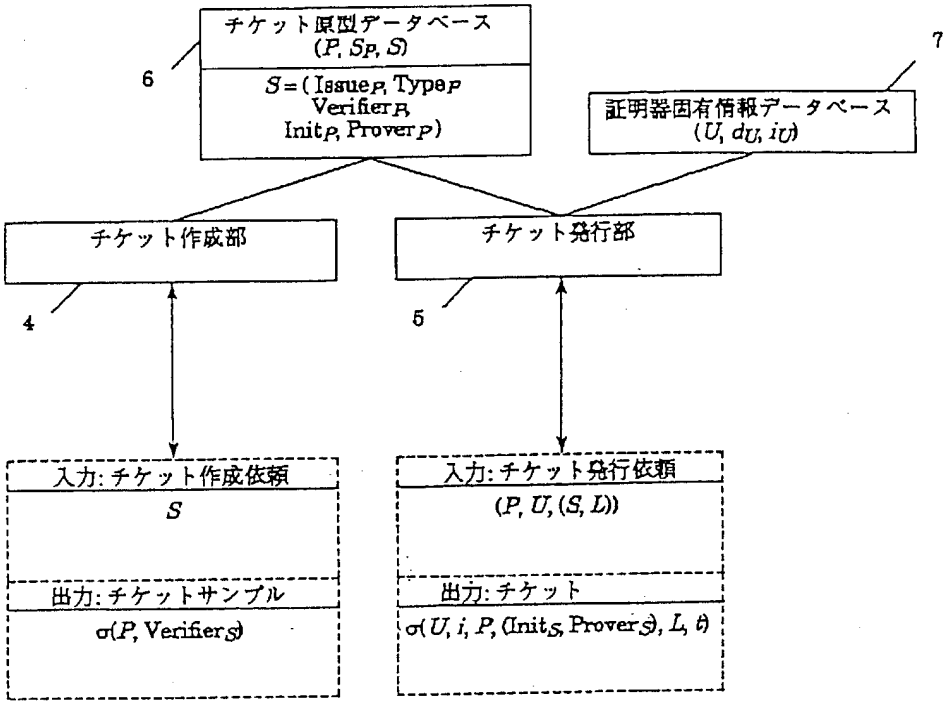
【図８】


開始
n に対応づけられた領域 i_n を確保できる?
81
no
NG
領域確保に失敗
yes
82
$i_n := 0$
OK

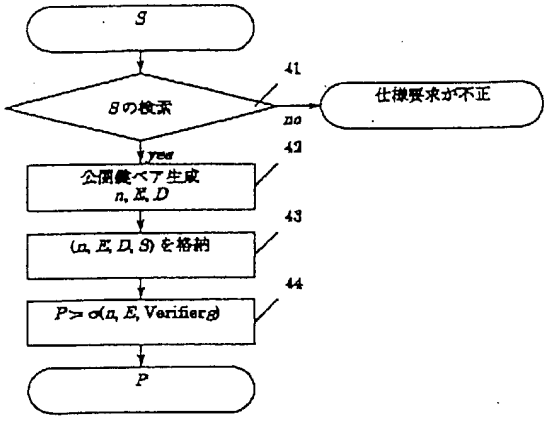
【図９】


開始
n に対応づけられた領域 i_n を確保できる?
91
no
NG
領域確保に失敗
yes
92
$i_n := L$
OK

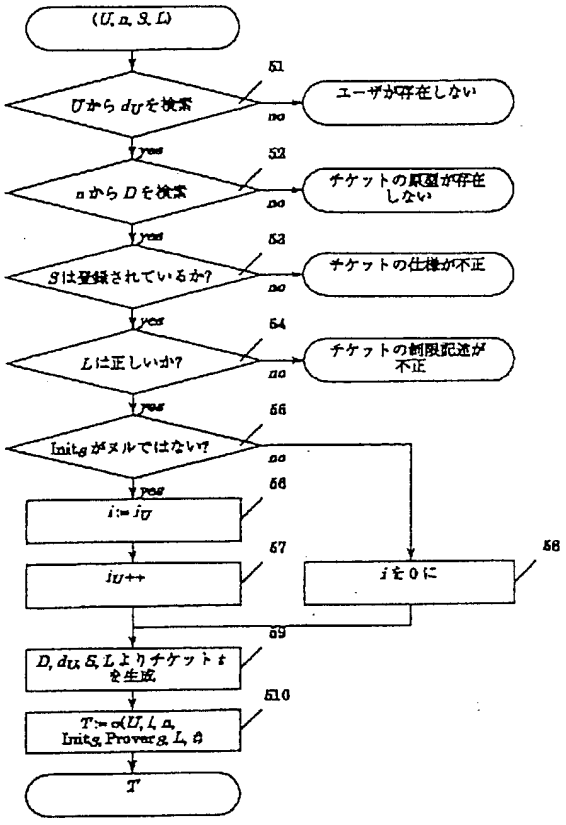
【図２】


チケット原型データベース
(P, SP, S)
S=(IssueP, TypeP
VerifierP,
InitP, ProverP)
6
7
証明器固有情報データベース
(U, dU, iU)
チケット作成部
チケット発行部
4
5
入力:チケット作成依頼
S
出力:チケットサンプル
σ(P, VerifierS)
入力:チケット発行依頼
(P, U, (S, L))
出力:チケット
σ(U, i, P, (InitS, ProverS), L, θ)

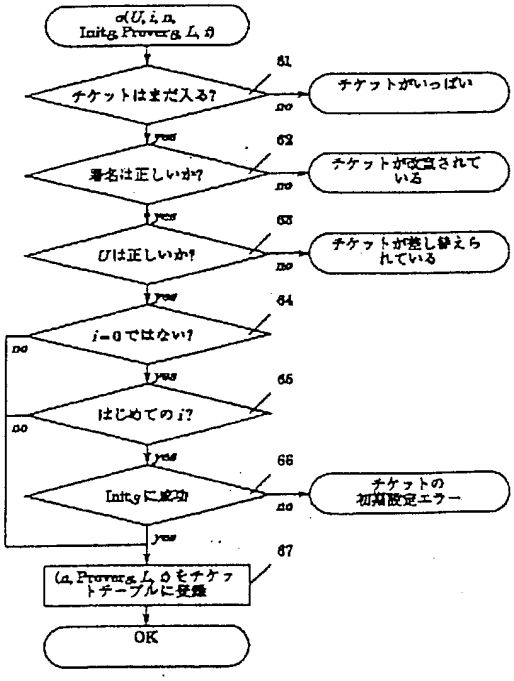
【図４】


S
41
Sの検索
no
仕様要求が不正
yes
42
公開鍵ペア生成
n, E, D
43
(n, E, D, S)を格納
44
P:= σ(n, E, Verifier_S)
P

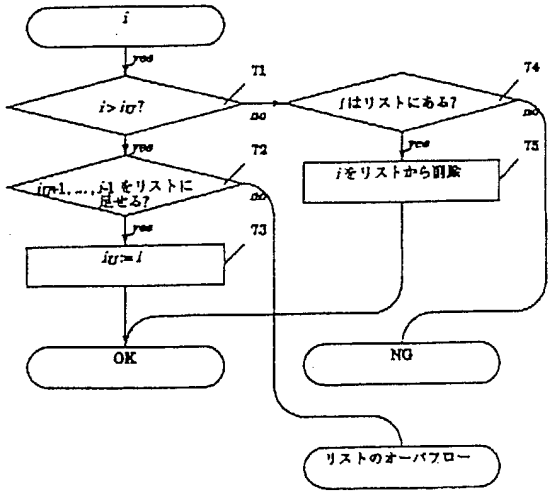
【図５】


(U, n, S, L)
51
Uから d_U を検索
no
ユーザが存在しない
yes
52
nからDを検索
no
チケットの原型が存在しない
yes
53
Sは登録されているか?
no
チケットの仕様が不正
yes
54
Lは正しいか?
no
チケットの制限記述が不正
yes
56
Init_S がヌルではない?
no
yes
56
i := i_U
57
i_U++
58
i を0に
59
D, d_U, S, L よりチケット t を生成
510
T:= σ(U, i, n, Init_S, Prover_S, L, t)
T

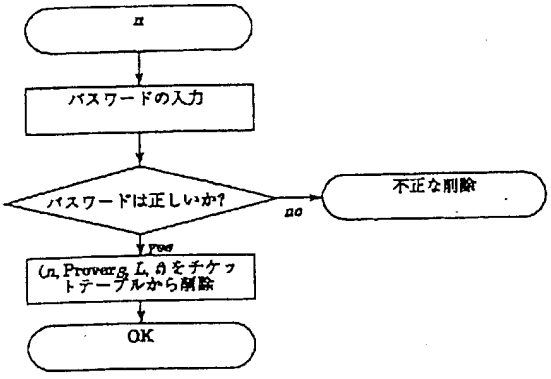
【図６】


σ(U, i, n, Init_S, Prover_S, L, S)
61
チケットはまだ入る?
no
チケットがいっぱい
yes
62
署名は正しいか?
no
チケットが改竄されている
yes
63
Uは正しいか?
no
チケットが差し替えられている
yes
64
i=0ではない?
no
yes
65
はじめての i?
no
yes
66
Init_S に成功
no
チケットの初期設定エラー
yes
67
(n, Prover_S, L, 0 をチケットテーブルに登録
OK

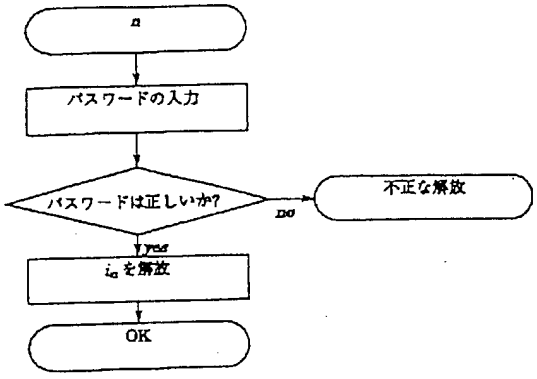
【図７】


i
yes
71
i> i_U?
no
yes
72
i_U+1, ..., i-1 をリストに足せる?
no
yes
73
i_U := i
OK
74
i はリストにある?
no
yes
75
i をリストから削除
NG
リストのオーバフロー

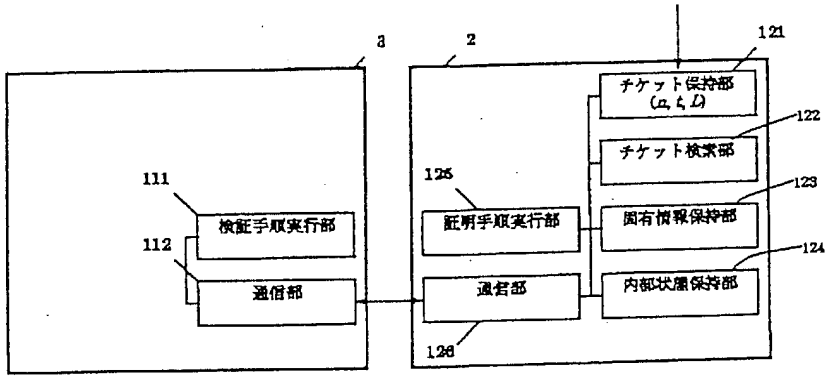
【図１０】



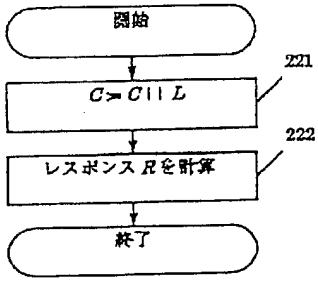
【図１１】



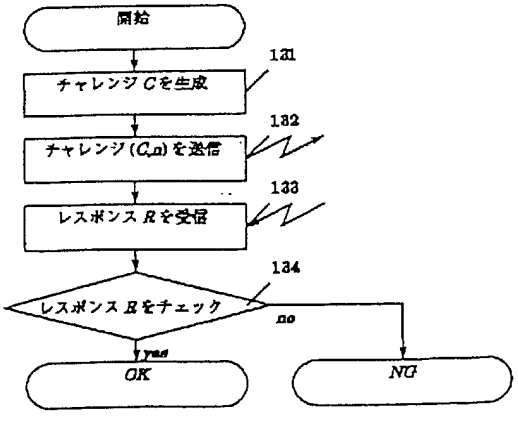
【図１２】



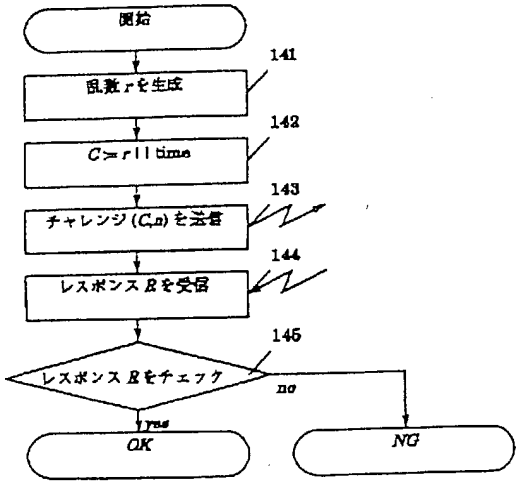
【図２２】



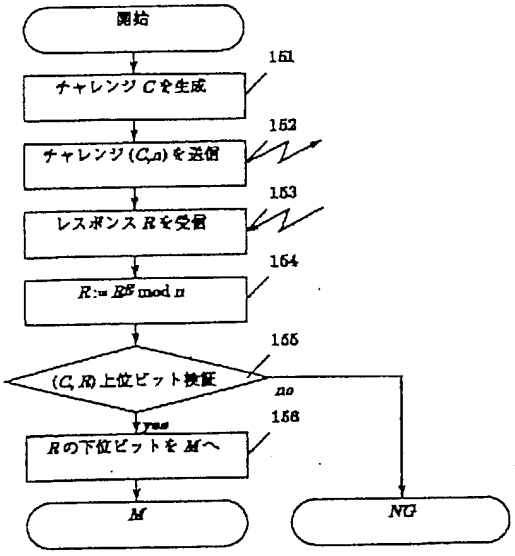
【図１３】



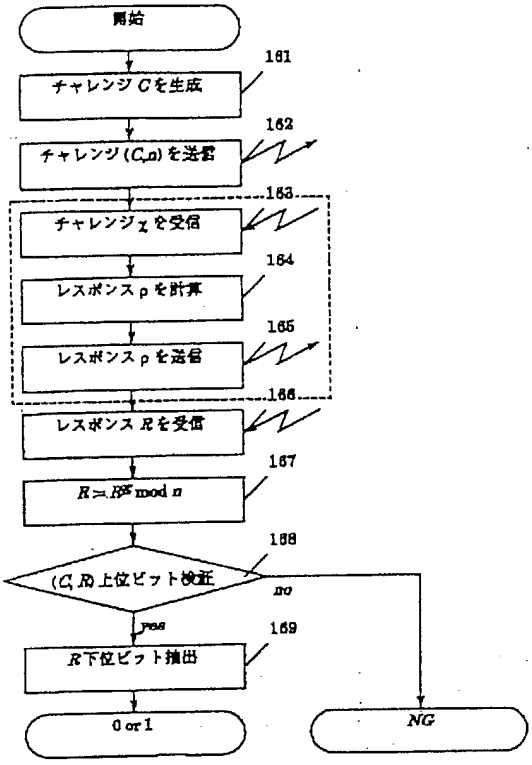
【図１４】



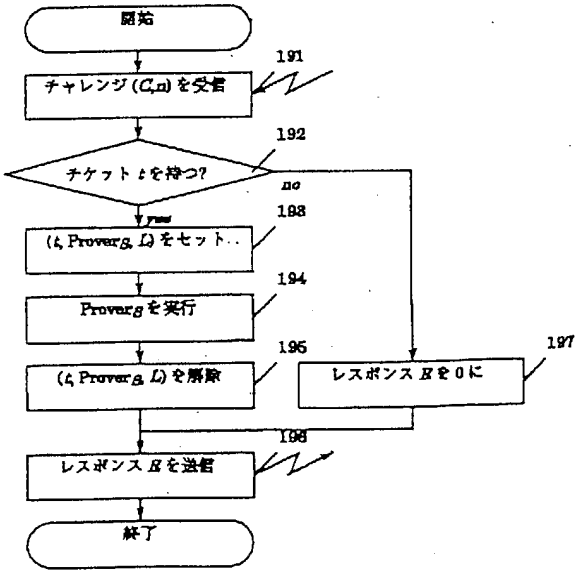
【図１５】



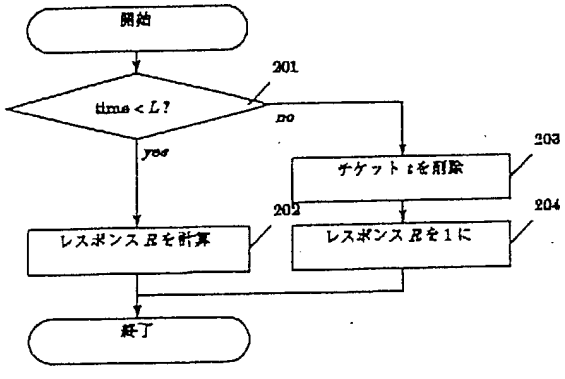
【図１６】



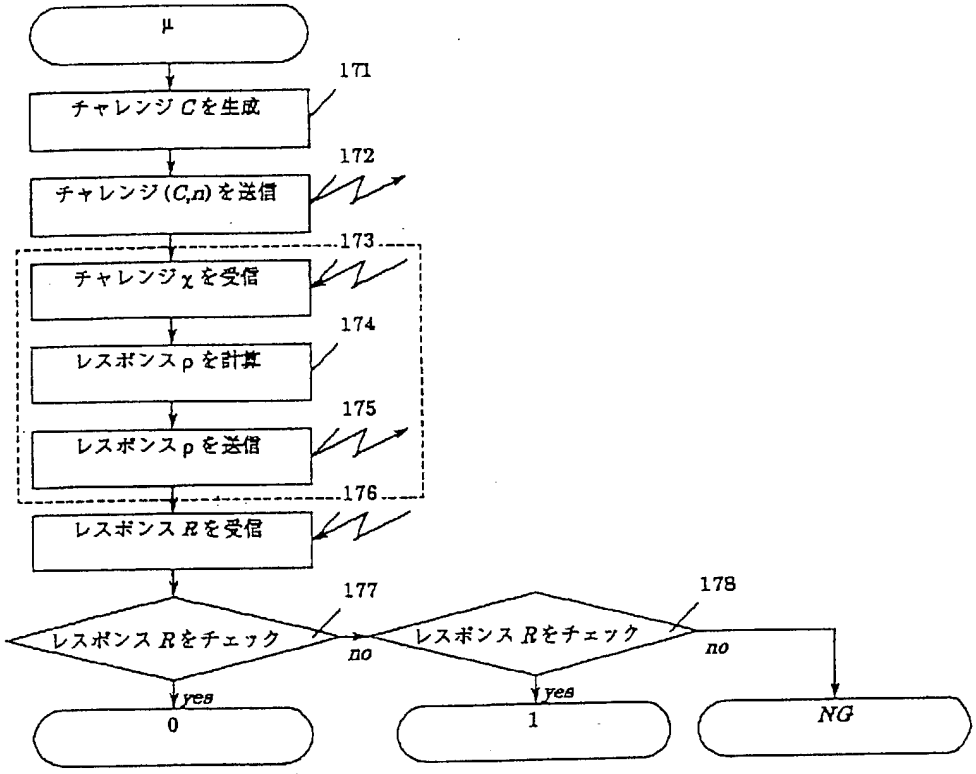
【図１９】



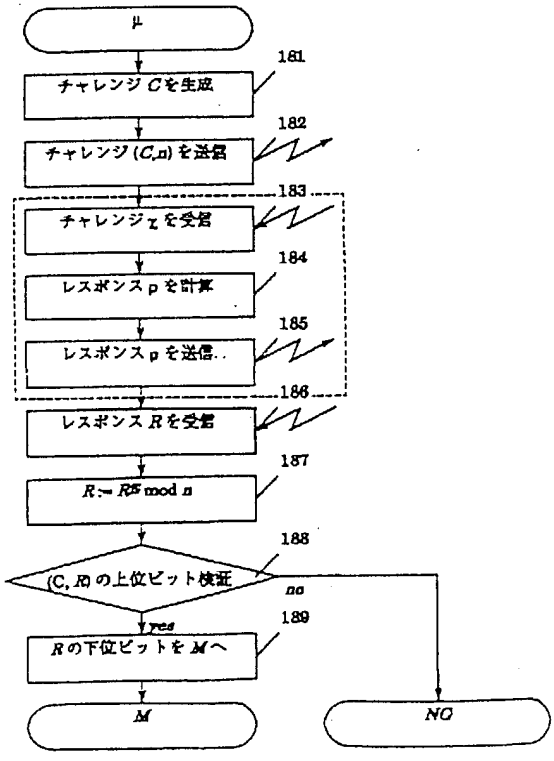
【図２０】



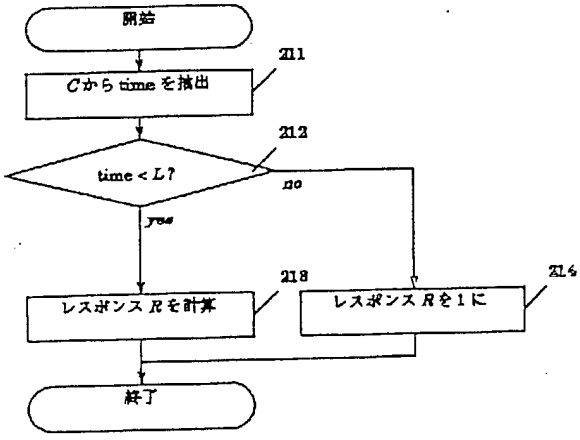
【図１７】


μ
171
チャレンジ C を生成
172
チャレンジ (C,n) を送信
173
チャレンジ χ を受信
174
レスポンス ρ を計算
175
レスポンス ρ を送信
176
レスポンス R を受信
177
レスポンス R をチェック
no
178
レスポンス R をチェック
no
yes
0
yes
1
NG

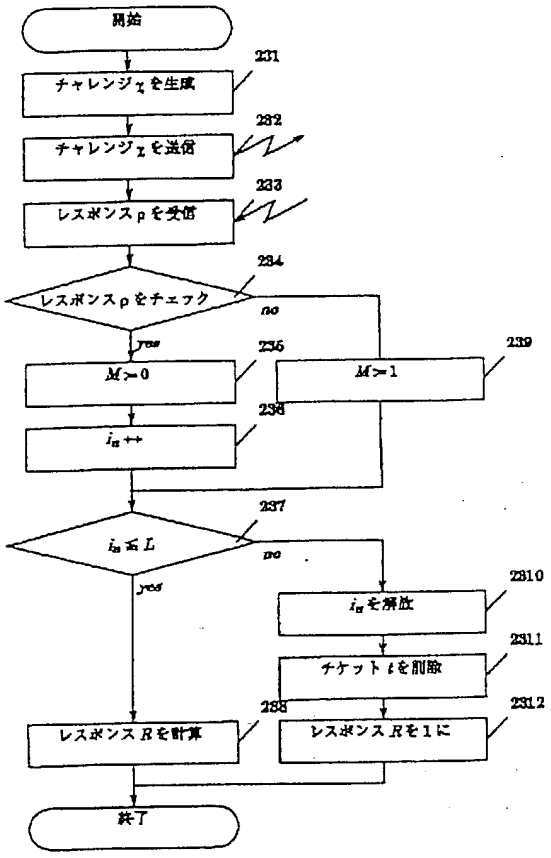
【図１８】


μ
181
チャレンジ C を生成
182
チャレンジ (C,n) を送信
183
チャレンジ χ を受信
184
レスポンス ρ を計算
185
レスポンス ρ を送信
186
レスポンス R を受信
187
R := R^E mod n
188
(C, R) の上位ビット検証
no
yes
189
R の下位ビットを M へ
M
NG

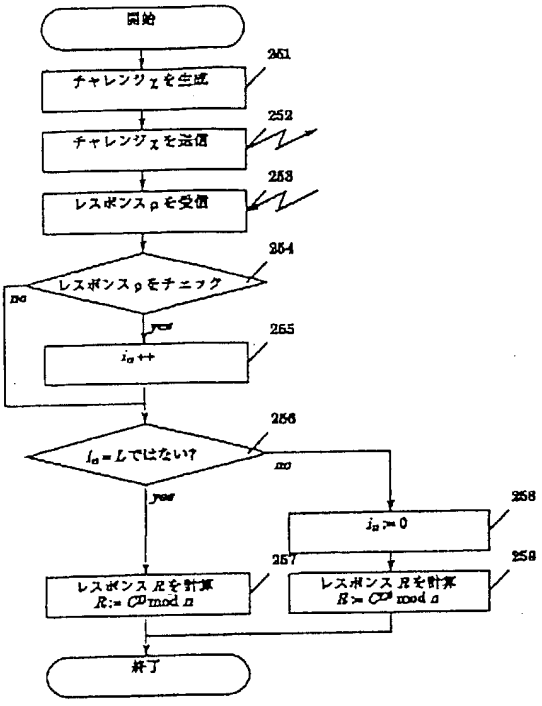
【図２１】


開始
211
C から time を抽出
212
time < L ?
no
yes
218
レスポンス R を計算
214
レスポンス R を 1 に
終了

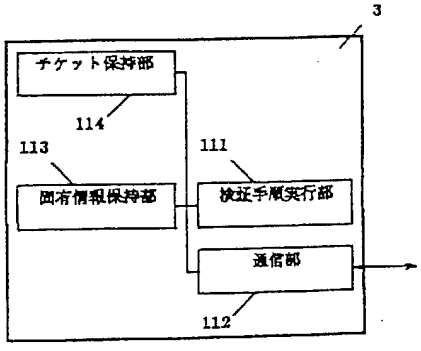
【図２３】



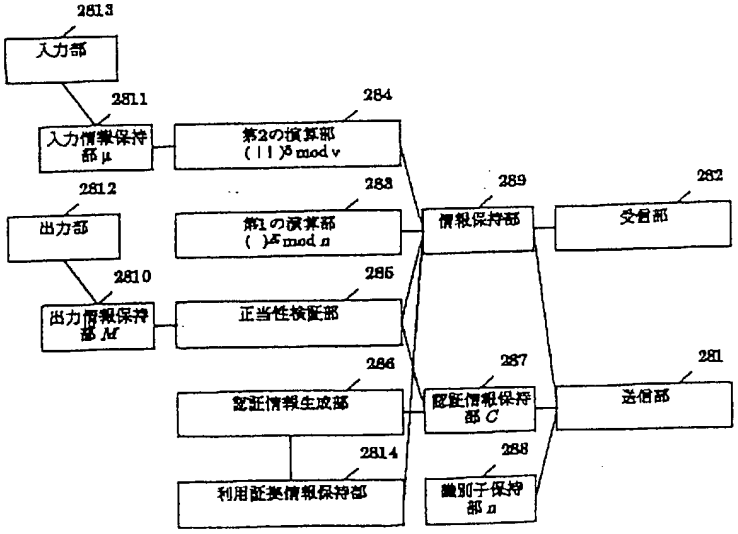
【図２５】



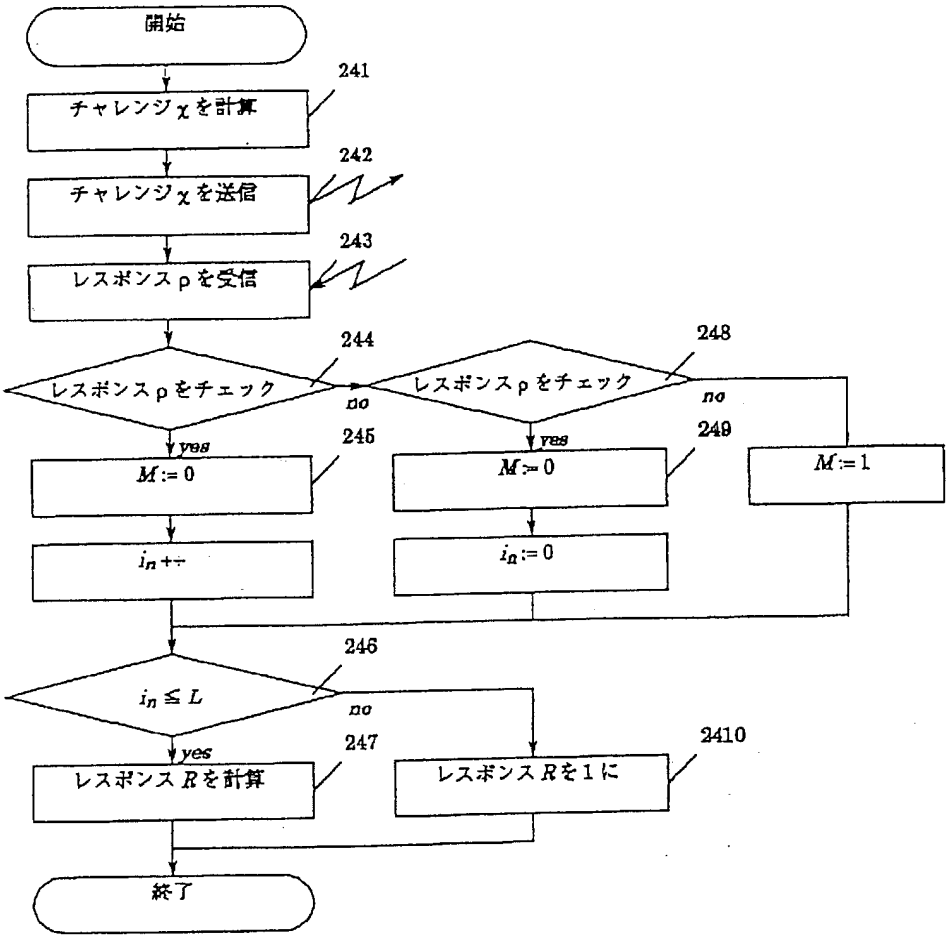
【図３０】



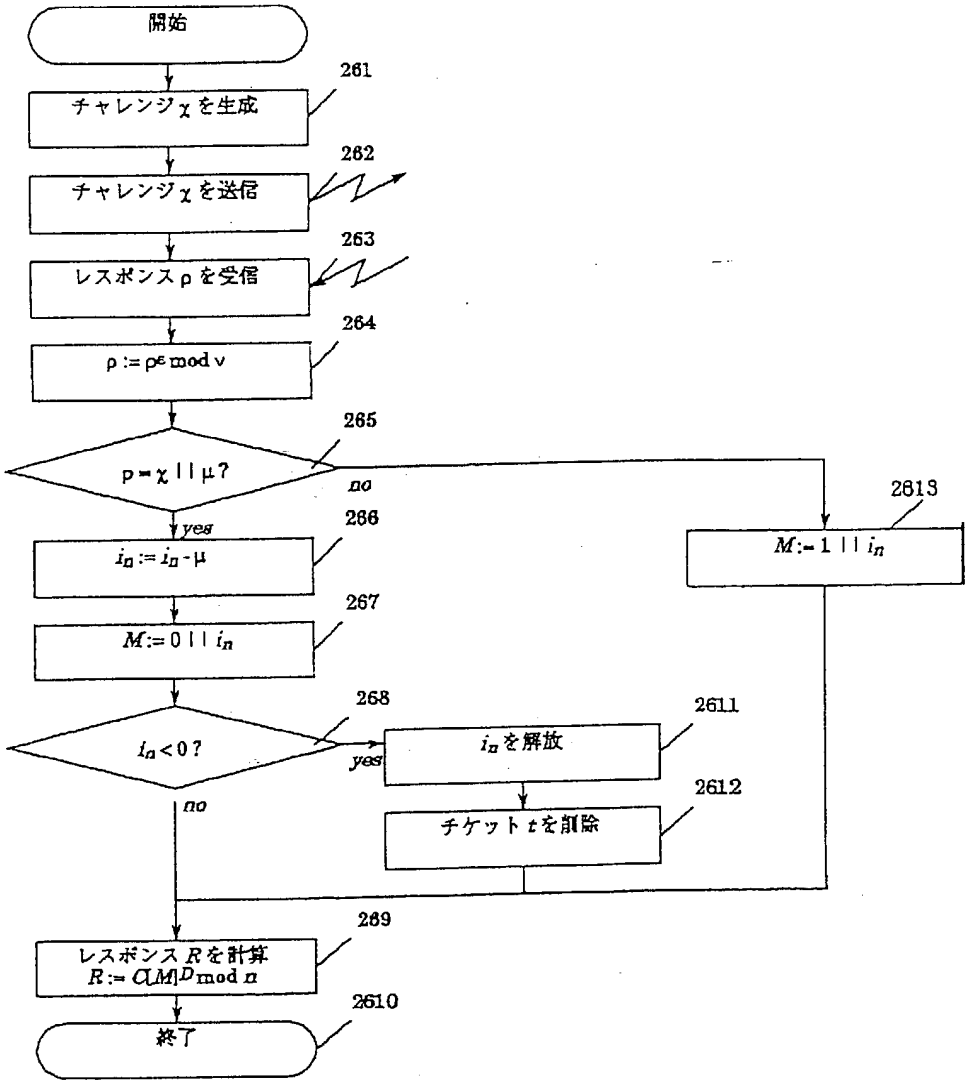
【図２８】



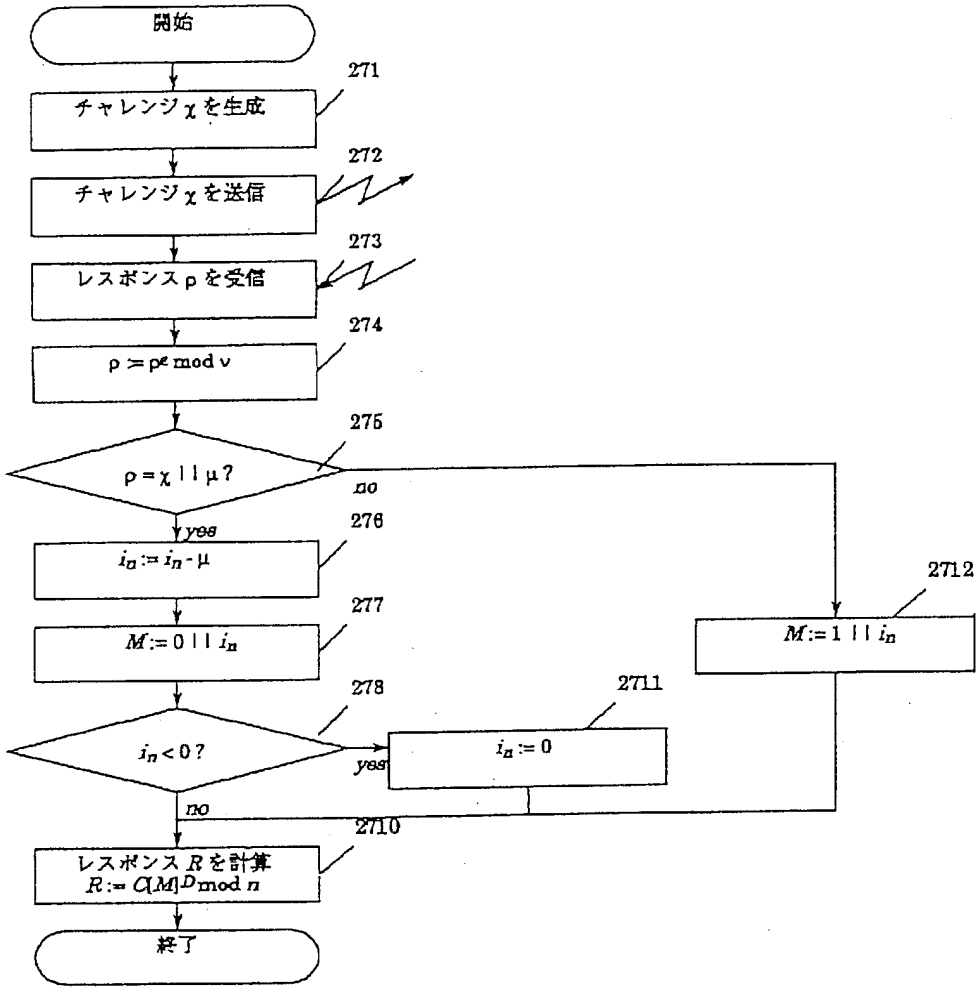
【図２４】


開始
241
チャレンジχを計算
242
チャレンジχを送信
243
レスポンスρを受信
244
レスポンスρをチェック
no
yes
245
M := 0
i_n ++
248
レスポンスρをチェック
no
yes
249
M := 0
i_n := 0
M := 1
246
$i_n \leq L$
no
yes
247
レスポンスRを計算
2410
レスポンスRを1に
終了

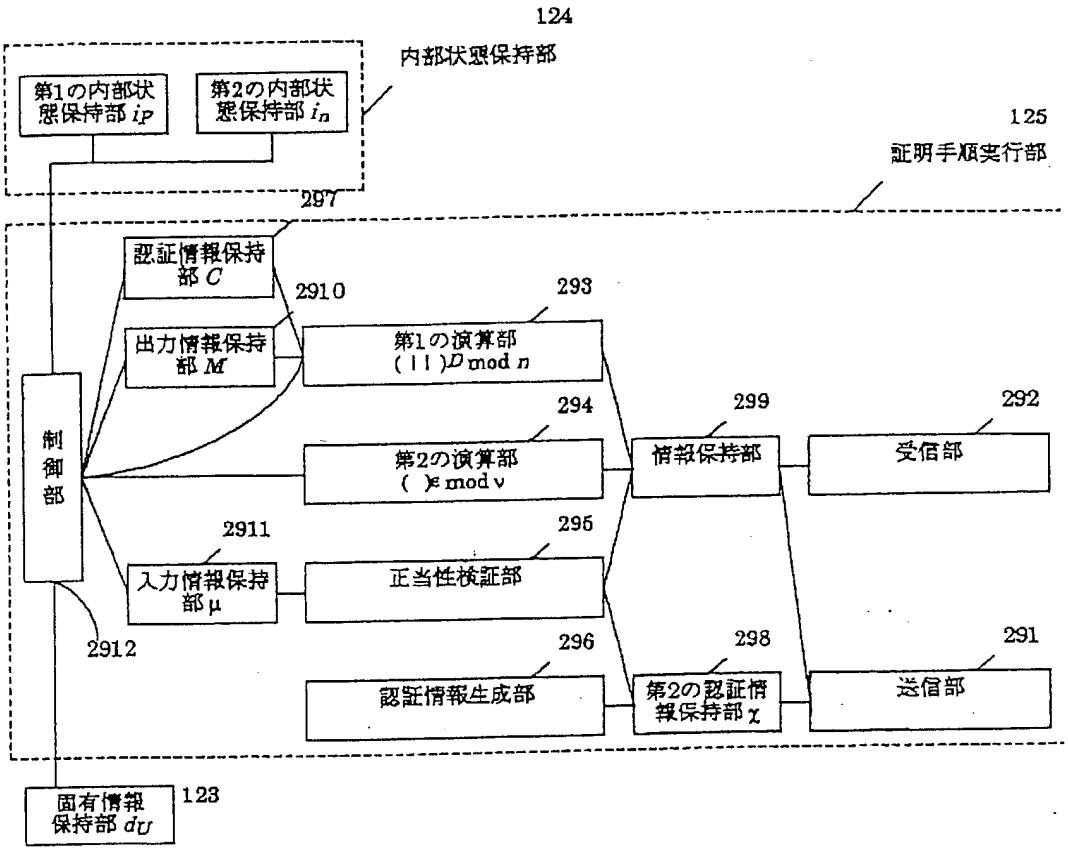
〔図２６〕


開始
261
チャレンジχを生成
262
チャレンジχを送信
263
レスポンスρを受信
264
ρ := ρ^ε mod v
265
ρ = χ || μ?
no
yes
2613
M := 1 || i_n
266
i_n := i_n - μ
267
M := 0 || i_n
268
i_n < 0?
yes
2611
i_n を解放
2612
チケットtを削除
no
269
レスポンスRを計算
R := C[M]^D mod n
2610
終了

【図２７】

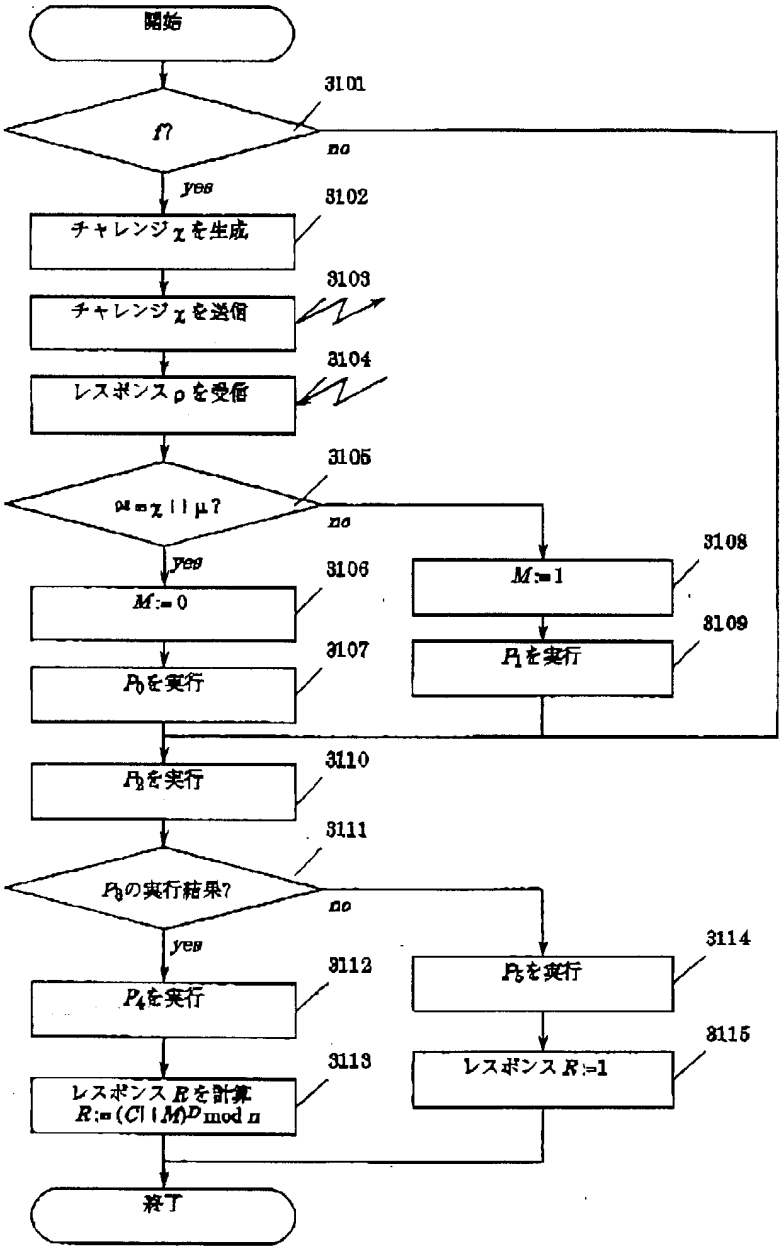

開始
271
チャレンジχを生成
272
チャレンジχを送信
273
レスポンスρを受信
274
ρ := ρ^e mod ν
275
ρ = χ || μ ?
no
yes
276
$i_n := i_n - \mu$
277
$M := 0 \,||\, i_n$
278
$i_n < 0$?
yes
2711
$i_n := 0$
no
2712
$M := 1 \,||\, i_n$
2710
レスポンスRを計算
$R := C[M]^D \bmod n$
終了

【図２９】

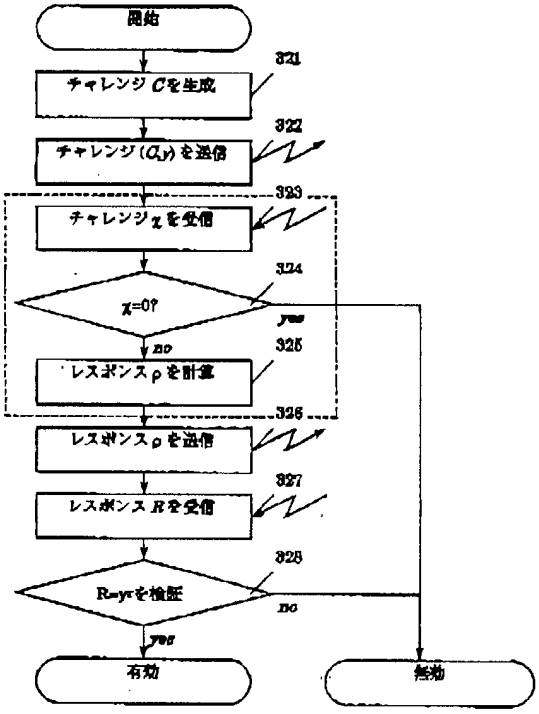

124
内部状態保持部
第1の内部状態保持部 i_P
第2の内部状態保持部 i_n
125
証明手順実行部
297
認証情報保持部 C
2910
出力情報保持部 M
293
第1の演算部 $(\ \ )^D \bmod n$
294
第2の演算部 $(\ )^e \bmod v$
299
情報保持部
292
受信部
制御部
2911
入力情報保持部 μ
296
正当性検証部
2912
296
認証情報生成部
298
第2の認証情報保持部 χ
291
送信部
固有情報保持部 d_U
123

1).

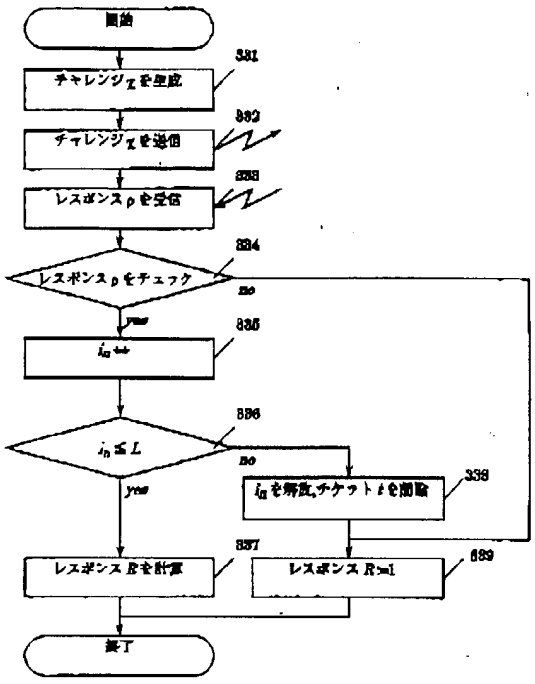
【図31】


開始
3101
f?
no
yes
3102
チャレンジχを生成
3103
チャレンジχを送信
3104
レスポンスρを受信
3105
ρ=χ||μ?
no
yes
3106
M:=0
3107
P0を実行
3108
M:=1
3109
P1を実行
P2を実行
3110
3111
P3の実行結果?
no
yes
3112
P4を実行
3113
レスポンスRを計算
R:=(C||M)^D mod n
3114
P5を実行
3115
レスポンスR:=1
終了

【図３２】



【図３３】



フロントページの続き

(51) Int. Cl. [6]	識別記号	F I	
H04L 9/32		H04L 9/00	675C